(1)

Bolt Beranek and Newman Inc.

[illegible]

Report No. 3896

SPEECH COMPRESSION AND SYNTHESIS

Quarterly Progress Report No. 1
6 April - 5 July 1978

July 1978

Prepared for:
Advanced Research Projects Agency

SPEECH COMPRESSION AND SYNTHESIS

Quarterly Technical Progress Report No. 1

6 April - 5 July 1978

ARPA Order No. 3515

Contract No. F19628-78-C-0136

Name of Contractor:
Bolt Beranek and Newman Inc.

Effective Date of Contract:
6 April 1978

Contract Expiration Date:
5 April 1979

Principal Investigators:
Dr. John Makhoul
(617)491-1850, x332

Dr. R. Viswanathan
(617)491-1850, x336

Sponsored by
Defense Advanced Research Projects Agency (DoD)
Monitored by RADC/ETC

TABLE OF CONTENTS

PROJECT PERSONNEL

John Makhoul	Principal Investigator
R. (Vishu) Viswanathan	Principal Investigator
Jared Wolf	Senior Scientist
John Klovstad	Senior Scientist
Lynn Cosell	Research Engineer
Richard Schwartz	Research Engineer
Dennis Klatt	Consultant
Victor Zue	Consultant
Peter Cudhea	Programmer
Kathleen Starr	Project Secretary

1. SUMMARY

During the past quarter, we have been working in the areas of natural phonetic speech synthesis and in real-time vocoder development. Our progress in these areas is summarized below and described more fully in Sections 2 (speech synthesis) and 3 (vocoder development).

Speech Synthesis

The goal of our work in speech synthesis is the development of a phonetic synthesis algorithm that results in natural-sounding speech quality. The input to the algorithm is a sequence of triplets, with each triplet comprising a phoneme and its pitch and duration. This synthesizer development is viewed as an initial step toward the design of a very low rate (75 bits per second) speech transmission system [Makhoul et al., 1977], but we feel that it will also find application in text-to-speech and voice message systems. The speech synthesis approach we have adopted is LPC synthesis from the concatenation of diphone templates. During the past quarter, we have designed and implemented the basic software for extracting diphone templates from natural speech utterances, forming a data base of diphone templates, and synthesizing new utterances by using stored diphone templates. We have also formed an initial limited data base of diphone templates and used this data base to synthesize our first sentences. The quality of the synthesized speech is high and it sounds quite natural. Section 2 is a detailed account of our progress in this area.

Vocoder Development

A second area of work in the past quarter has been real-time vocoder development. The goal of this effort is the real-time implementation of several speech analysis and synthesis algorithms that have the promise of producing improved speech quality at average data rates of about 2000 bits per second (not including silence). The starting point for this project is the real-time LPC-II vocoder program implemented on the Floating Point Systems Inc. (FPS) AP-120B array processor at the Information Sciences Institute (ISI). During this quarter, we have devoted considerable effort to bringing up on our system the EPOS PDP-11 operating system, the PDP-11 vocoder control program, and the AP-120B vocoder speech analysis/synthesis program, all developed by ISI. We have run real-time tests of the vocoder, both on the BBN machine alone and in a vocoder conversation session with ISI over the ARPA Network. At present, the quality of the vocoded speech is not up to expectations, and there seems to be a timing problem in our PDP-11/40 host computer. With the aid of ISI, we are investigating these problems.

In addition, we have studied the structure and details of the ISI AP-120B and PDP-11 programs, and we have developed a plan to make them more modular, thus facilitating the addition of the new algorithms. Section 3 describes our work in these areas.

We attended the 1978 IEEE International Conference on Acoustics, Speech, and Signal Processing, April 10-12, 1978, in Tulsa, Okla., and we presented three papers on our speech compression work. These papers, which are included in the final report of the previous speech compression contract [Viswanathan et al., 1978a], are listed in the references [Makhoul and Viswanathan, 1978; Makhoul et al., 1978; Viswanathan et al., 1978b].

BBN was the host of the NSC project meeting on May 22-23, 1978. At that time, our new speech compression and synthesis contract had just begun, and so we reported only on our plans for this work. However, we also described related work on wideband speech coding and the enhancement of wideband noise corrupted speech, sponsored by other contracts.

2. PHONETIC SPEECH SYNTHESIS

During the past quarter we have designed and implemented a first version of most of the software needed for the speech synthesis project. In this section we discuss, in detail, some of the design decisions that were made and the phenomena that necessitated them.

Section 2.1 describes our basic approach to diphone synthesis. Reasons for choosing the diphone as the basic unit of synthesis, along with a detailed description of our use of diphone templates are presented.

In Section 2.2 we discuss some of the extensions to the definition of a diphone that we feel are necessary. The acoustic-phonetic phenomenon necessitating each extension is first given, followed by a description of the particular extension that resolves this difficulty and a discussion of the implications of these solutions on the design goals of the project.

Section 2.3 contains a summary of the major programming efforts thus far. The functions of the diphone template compiler (COMPOSE) and the program that concatenates diphones and synthesizes speech (SPEAK) are described in detail.

During this initial quarter, most of our experiments have been for the purpose of debugging and testing our basic assumptions about the nature of the diphone synthesis task. Three preliminary experiments are described in Section 2.4.

Section 2.5 contains a further discussion on the effects of our design decisions on an eventual goal of producing a phonetic vocoder. It also discusses the possibility of vastly reducing the storage requirements for diphone templates.

Section 2.6 lists the major efforts for the remainder of the project, which are:

1) the design, acquisition, and use of a data base for identification and extraction of all the necessary diphones
2) development of algorithms that automatically generate pitch and gain tracks from a single value of pitch and duration (and the phoneme string).
3) further improvement of the quality of the speech produced by the program.

2.1 Synthesis Methodology

2.1.1 Diphone Speech Synthesis

A string of phonemes to be synthesized is first translated into a corresponding diphone sequence. A diphone is defined as the region from the middle of one phoneme to the middle of the next phoneme. Diphone synthesis is accomplished by concatenating diphone templates, which are then used to synthesize speech. A diphone template consists of the LPC parameters necessary for synthesizing one diphone. To produce natural-sounding speech, sentence inflection is determined by extracting the duration and one value of pitch from each phoneme from a real utterance of the same sentence.

The diphone is a natural unit for synthesis because the coarticulatory influence of one phoneme does not usually extend much further than half way into the next phoneme. Since diphone junctures are usually at articulatory steady states, minimal smoothing is required between adjacent diphones. Since the regions around the phoneme transitions are preserved intact, the difficult task of duplicating these transitions by complicated acoustic-phonetic rules is avoided. If approximately 40 to 50 phonemes are assumed, there are roughly 1600 to 2000 diphones in English. Many of the diphones that are not allowed within a single word will appear across two words. Since the length (duration) of the diphone templates will not generally be the same as the duration required in synthesis, the template is lengthened or shortened before being concatenated with other diphones. Also, to avoid discontinuities, the spectral parameters are smoothed where two diphone templates abut.

2.1.2 Diphone Specification

To avoid restricting ourselves so early in our research, we have defined the diphone template in a very general way. A text file contains all the specifications of all the templates. This file identifies the diphone and specifies the sentence from which the diphone template is to be extracted. The template, which will be extracted during a compiling phase by COMPOSE (see Sec. 2.3.2.1), consists of the parameters within the region of this sentence, as specified by two frame numbers (time specifications). The specification also includes the location of the phoneme boundary and two interpolation points. These interpolation points specify the region that is to be interpolated with the adjoining template to smooth out the discontinuity at each diphone boundary.

The diphone templates are then assembled by using a compiler (see Sec. 2.3.2.1) that collects the parameters in a single diphone dictionary file. Each template contains 14 log area ratio parameters, a value of pitch, and the gain for each frame of speech. Since the average diphone is about 10 frames (10 ms/frame) long, there are about 160 values stored for each template.

2.2 Extending Diphone Applicability

In this section, we describe some extensions to the fundamental diphone definition stated above. These extensions allow "special cases" to be handled (thus permitting the highest possible intelligibility and naturalness) uniformly by the synthesis program.

2.2.1 Splitting Phonemes

So that the same templates can be used for all phonemes, some phonemes that have more than one acoustically distinct region have been split up into two "pseudophonemes".

The unvoiced plosives [P,T,K] are each made up of a region of silence followed by a burst and aspiration region. The affricates [CH,JH] are similar in that they also have two distinct regions. Since it is important that we do not perform smoothing of parameters across the burst, and since the durations of the two halves of the unvoiced plosive phonemes vary independently, we have chosen to define all of these phonemes by a pair of pseudophonemes. The first pseudophoneme, "SI", is for the silence region, and it is followed by a second pseudophoneme telling which plosive it is. Thus, what would normally be represented phonemically as [P] is now a sequence [SI-P].

The diphthongs also have two very different acoustic regions. The diphthong [AY], as in "bite", for instance, starts out much like an [AA], as in "pot", but ends up more like the [IY] in "beet". The two relatively steady regions are delineated by a rapid transition between them. As with the unvoiced plosives, the durations of the two steady regions are somewhat independent, as are the contextual effects of neighboring phonemes. Therefore, some of the diphthongs have been artificially split into two pseudophonemes (e.g., [AY1,AY2]), which will appear only in sequence.

2.2.2 Context-Specific Diphones

Diphones are usually independent of context, but there are important exceptions. For instance, the unvoiced plosive [T] is changed greatly when followed by an [R], and in the sequence [W-IH-L], as in the word "will", the entire [IH] phoneme is drastically affected by the presence of the [L]. Consequently, there must be a separate diphone template for [W-IH] to be used in this context. We have therefore allowed more than one template to be defined for diphones when they are affected by context in this way. An example is given in the following notation, where

W IH / & L

specifies the diphone [W-IH] with the right context of [L]. The "/" means "in the context of" and the "&" is a place marker for the W-IH pair. The context can be specified for either or both sides, and the context at any one phoneme can be a list of phonemes rather than a single phoneme. For instance, the sequence

SI K / S & [AH AO AA UW R L]

is used to specify a SI-K sequence preceded by S and followed by back vowels or glides. These context effects are relatively infrequent, so most diphones have only a single template.

2.3 Current Implementation

The programs implemented during this quarter fall into two categories:

a) Programs used for research and development of suitable diphone templates,

b) Programs that organize these templates and use them for synthesis.

2.3.1 Diphone Acquisition

2.3.1.1 Display Programs

Our general signal processing and display program was modified so that a user can interactively edit the manual phonemic transcriptions associated with a sentence. The program displays 10 time-varying parameters (such as energy and formants) along with user-defined manual transcriptions. The user can also define diphone templates (which are appended to the text file of diphone templates) by interactively marking the diphone frame boundaries and the interpolation points.

An addition has been made to the Acoustic-Phonetic Experiment Facility (APEF) [Schwartz, 1976] that allows the user to display simultaneously all the occurrences of a particular diphone. The user specifies at run time which parameters are to be plotted on the display and how much context is desired. This display facility aids the user in deciding which of several occurrences of a diphone is most typical and would be the best to use as a diphone template.

PRPLOT, which produces high resolution hardcopy (Calcomp plotter) displays, was improved so that it can also plot several parameters for short portions of sentences. It can also mark the frames chosen for diphone boundaries. This program will be used, in a manner similar to the APEF display program, to facilitate comparisons of different occurrences of the same diphone. The advantage of using this program, rather than the display version, is that the resolution is greater and it is possible to automate the plotting.

2.3.1.2 Semiautomatic Diphone Specification

To expedite the diphone acquisition process, we have begun to experiment with a semiautomatic procedure for specifying the frame boundaries in a diphone template. A program has been written that transforms the manual transcription of a sentence (including phone boundary times) into a set of diphone specifications. Currently, the frame nearest the middle of the phoneme is defined to be the diphone boundary. The interpolation points are calculated to be a percentage of the distance between the phoneme boundary and the diphone boundaries. This percentage can depend on the two phonemes involved. For instance, the interpolation regions for consonant pairs probably needs to be wider than for vowel pairs.

Preliminary experience with this semiautomatic procedure is encouraging, in that it seems to give results comparable to those derived by hand.

2.3.2 Synthesis-Related Programs

2.3.2.1 COMPOSE

COMPOSE is a program that "compiles" a working set of diphone templates into a form that the synthesis program SPEAK (see 2.3.2.2 below) can conveniently use. It takes as input a text file of diphone definitions. Each diphone defined in this text file has at least one and possibly several entries, depending on its contextual variability. Each entry specifies one (and only one) diphone template. Although any diphone may have many different templates (one for every significantly different context in which it might occur), it is required that every diphone have one template specified with no context, which is to be used whenever all of the specified contexts are inappropriate. A typical diphone entry consists of:

1) A pair of phonemes that (taken together) name the diphone being defined;

2) Left and/or right phonetic context in which this diphone template is appropriate if necessary;

3) The name of the parameter file from which the parametric information necessary for the synthesis of this diphone template is to be extracted;

4) The five frame numbers (two diphone boundaries, two interpolation points, and one phoneme boundary) that define the portion of the parameterized utterance that is to be extracted and used as the diphone template.

Figure 1 shows a short segment of a current diphone definition text file.

```
SI K
  117  118  119  120  120 <5SPEECH>JJW103

SI K / & [ AO AA R ]
  49   50   51   52   53 <5SPEECH>JJW122

SI P
  116  117  119  119  119 <5SPEECH>JJW123
```

Fig. 1. Excerpt from a diphone definition text file.

COMPOSE reads the diphone definition text file, processing it incrementally one entry at a time. For each entry, it checks to see that all of the phonemes used are defined, opens the file containing the parametric information, reads all of the necessary parametric information (14 log area ratios, fundamental frequency, and gain), and creates an entry for this diphone (by the specified context) for subsequent references (by the program SPEAK).

2.3.2.2 SPEAK

SPEAK, the phonemic speech synthesis program,

1) takes as input a sequence of phonemes, with their corresponding durations, pitch, and gain;

2) expands the phoneme sequence into a diphone sequence;

3) selects the most appropriate diphone template (depending on the local phonetic context) from among those in its diphone definition file;

4) time-warps each of the diphone templates (to realize the specified durations);

5) interpolates between consecutive warped diphone templates to minimize the perceptual effects of spectral discontinuities;

5) uses the resulting sequence of log area ratios (specified every 10 ms) in combination with the pitch and gain information supplied as input to control an LPC speech synthesizer.

Diphone Selection -- After SPEAK has read in the phonemes, duration, pitch, and gain of the utterance to be synthesized, it takes consecutive phonemes two at a time and determines the corresponding diphone. If the appropriate diphone definition is not currently defined, the user is informed of the problem and given an opportunity to substitute a similar diphone that is already in the definition file. A search is made through the templates that are associated with the designated diphone to find the one with the most constrained context satisfied by the phonetic context of the sentence. Since every diphone definition must include at least one template that is to be used as a default (in the absence of suitable contexts), the synthesis can be completed regardless of the phonetic context.

Diphone Warping -- Experiments have shown that when the rate of speech changes, most of the rate change occurs during the middle parts of the phonemes. Furthermore, the rate of spectral change during phoneme transitions remains relatively invariant. Consequently, our current diphone warping algorithm treats the middles of the phonemes as being much more elastic than the phoneme transitions when adjusting the duration. The region defined as less elastic is a fixed fraction (program constant) of the noninterpolated portion of the phoneme around the phoneme boundary. This constant is typically 0.5 (i.e., typically 25% of the whole phoneme). Another program constant determines how inelastic this region is to be.

Diphone Interpolation -- There may be a discontinuity in the log area ratio parameters where two diphone templates abut. These discontinuities are eliminated by replacing the log area ratios between the two interpolation points adjacent to this diphone boundary by interpolated log area ratios. Figure 2 illustrates this operation. This interpolation is done for each log area ratio. Interpolation also compensates for some of the coarticulation effects of phoneme adjacency since it smoothly combines the log area ratio parameters from the two diphones in the region of the boundary.

Speech Synthesis -- The speech synthesis program used is the same program used in our low rate LPC speech compression effort.

2.4 Current Results

While most of this quarter was spent in program design and development, we were also able to perform some initial experiments.

2.4.1 Self-Synthesis

We conducted a self-synthesis test that involved synthesizing two sentences, each with diphone templates that had

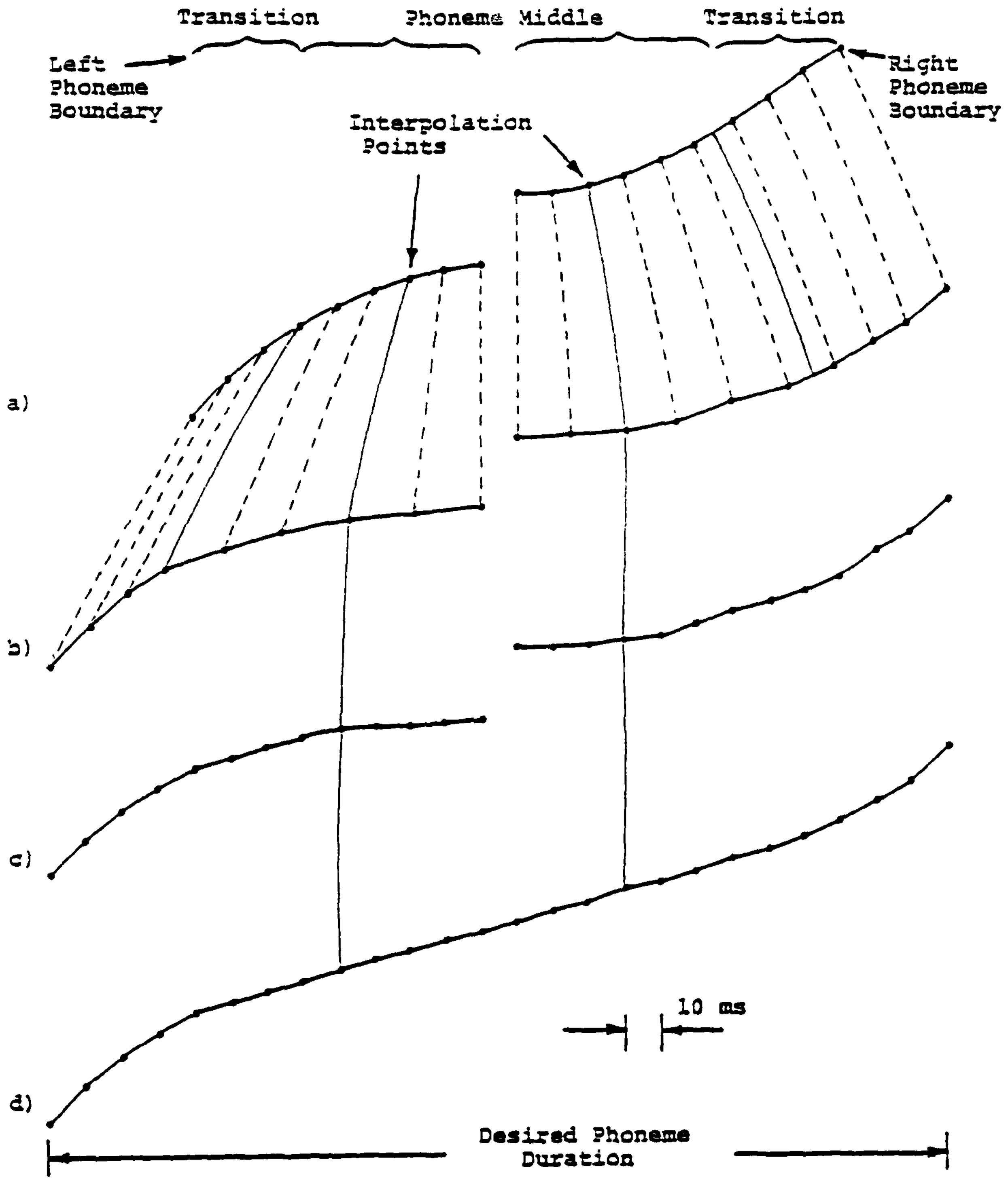

Fig. 2 Time warping and interpolation of one parameter.
a) Original templates showing interpolation points and partition into phoneme-middle (more elastic) and phoneme-transition (less elastic) regions, b) Parameter time-warped to fit the desired phoneme duration, c) Parameter resampled to 10 ms frames, d) Final form of the parameter, after interpolation.

been extracted from the original sentence. This synthesis helped us test the programs COMPOSE and SPEAK, since the only difference between this process and normal LPC analysis-synthesis was that the log area ratios were interpolated through the middles of the phonemes. After a short period of debugging, the synthesized utterances were identical in every perceivable respect to the original vocoded speech.

2.4.2 Different Speaker

We also synthesized the voice of a new (male) speaker. The new speaker said the same sentences that had been synthesized in the experiment described above. When his speech was manually transcribed, it was found to contain nearly all of the same diphones for which templates had been extracted from the same sentence spoken by our model speaker. The phonemes, durations, and all pitch and gain information came from the new speaker. The synthesis was done as before, where the diphone templates came from the sentences spoken by our model speaker. The only exceptions were the one or two new diphone templates per sentence (necessitated by the new speaker's pronunciation), which had to be extracted from his own sentence. The synthesized speech in this case was very intelligible, but had definite characteristics of both speakers. However, as hoped, the characteristics of the new speaker (as conveyed in the pitch, gain, and duration information) were definitely the dominant ones. The speech was informally judged to be as natural as LPC vocoded speech.

2.4.3 Speech Rate Experiment

Since a fundamental part of the template concatenation process involves the time-warping of the diphone templates, we reproduced the results of the previous two experiments at a different speaking rate than the one at which the original sentences had been spoken. The different speaking rates were realized by modifying the original phoneme durations by a constant factor. We synthesized several sentences, using duration factors that ranged from Ø.7 to 1.5. The sentences synthesized with a factor of Ø.7 were quite intelligible, but unnatural in the sense that people rarely talk that fast. The sentences produced with a duration factor of 1.5 were also quite intelligible, although at a very slow speech rate.

2.5 Discussion

2.5.1 VLR Compatibility

In both examples of diphone extensions (Sec. 2.2), the addition of new phonemes and diphone context does not actually result in more diphones to be transmitted. This fact is important for the use of this system in a very-low-bit-rate (VLR) compression system. It is unnecessary to transmit the [SI] phoneme, since it is implied by the [P] that follows, as the [AY1-AY2] sequence is implied by sending the [AY]. However, in each of these two cases, it may be necessary to transmit two separate durations, requiring an extra 2 bits or so. The context-specific diphones discussed in Sec. 2.2.2 are automatically selected by the synthesis program from the transmitted phoneme sequence, and therefore do not require any additional information to be transmitted.

2.5.2 Data Reduction

At this early point in the project, we have chosen to preserve all frames of data in the template. To be sure that data reduction at this early phase does not make good results more difficult to attain, we do not quantize the parameters. Once we have achieved the desired results, we will investigate ways to minimize the diphone template storage. We anticipate that each template will then require no more than 200 bits of storage. Our variable frame rate algorithm allows us to use only 3 to 4 spectral frames per template. Each spectral frame can then be quantized to 50 bits. These bits would not be transmitted in a speech compression system, but would merely be data used by the synthesis program to reconstruct the diphones (indirectly reconstructing the phoneme sequence) during speech synthesis.

2.6 Future Work

2.6.1 Diphone Acquisition

Most of the necessary programs have now been written and debugged. Therefore, the remaining effort will be devoted primarily to accumulating the diphone templates, which, when warped, abutted, interpolated, and synthesized with an LPC synthesizer, will result in speech comparable to that achieved by using an LPC speech compression system alone. The completion of this effort depends primarily on the person (or program) that will specify the diphone templates and his indication (by choice of boundaries and interpolation points) of how they are to be used by the synthesis programs. This task will necessitate the manual transcription of several hundred sentences and the choice of a set of diphones to be used as templates.

2.6.2 Pitch and Gain Derivation

One goal of our speech synthesis project is to synthesize speech by using only a single value of pitch for each phoneme (and no gain information). In the experiments described above, we extracted detailed pitch and gain information from the sentence being synthesized. We will, therefore, need to experiment with different methods of generating the gain and pitch tracks. The gain tracks will be taken from the templates, with some shifting and/or smoothing. The pitch tracks will be generated from a single pitch value given for each voiced phoneme - possibly in conjunction with pitch contours in the templates.

3. REAL-TIME VOCODER DEVELOPMENT

In this section, we describe the work performed this quarter in bringing up the EPOS operating system and the LPC-II vocoder program, both developed at ISI, on our PDP-11/40 and AP-120B system. We also developed a plan for incorporating our new speech compression algorithms in the real-time vocoder, and we have implemented several pieces of support software.

3.1 Vocoder Installation

Although we encountered several unforeseen problems in attempting to install the ISI vocoder at BBN, we managed to make significant progress this quarter toward the realization of the real-time network vocoder. We obtained and installed the required hardware, collected the software, and configured it to run on our system. We would like to acknowledge ISI's cooperation and help in obtaining and bringing up the software.

3.1.1 AP-120B and Additional Memory Installations

In January 1978, Floating Point Systems, Inc. installed an AP-120B Array Processor on our PDP-11/40. We installed the AP software under RT-11 on the PDP-11 and tested the system. We also purchased and installed an additional 64K of MOS memory for the PDP-11. This additional memory is required to run EPOS, the ISI-developed network operating system that supports the vocoder.

3.1.2 Configuration of the ISI Vocoder for BBN System

Several changes to the vocoder program were required to accommodate differences between the ISI and BBN systems. The most significant difference is that ISI uses A/D and D/A converters interfaced directly to the AP-120B Array Processor, while BBN's converters are interfaced to the SPS-41. ISI had initially used the SPS-41 for analog/digital conversion, but for later versions of the vocoder, they switched to the AP-120B. This switch required identifying and changing various routines that use the converters in the current version. EPOS modifications were also necessary to allow access to the SPS-41.

An unanticipated difficulty arose over the support software. ISI performs all assembling and linking on their TENEX system, and our PDP-11 system would not accept the formats of the source files used by ISI. In addition, they had modified some of the FPS support software, which also ran under TENEX. We were forced to modify all of this support software to run under the TOPS-20 operating system.

Most of the problems encountered were caused by the differences in the system configurations. Since we will have a debugged version of the vocoder that uses the SPS-41 converters, as well as the current ISI version that uses AP-120B converters, the next site to bring up the vocoder on the AP-120B should not encounter the same problems. Because we also have both TOPS-20 and TENEX versions of the support software, any PDP-10 site should be able to use the software immediately.

3.1.3 Back-to-Back Test

Using FUD, an ISI-developed debugger for the AP-120B, we tested the AP-120B code for the vocoder. This test did not use the ARPA Network, but merely fed the parameters obtained from the analysis portion of the AP-120B program back to the synthesis portion of the program. We obtained moderately intelligible output speech in real-time. We were also able to store analysis parameters on a PDP-11 data file and subsequently transmit that file to ISI, who performed the synthesis. Using this testing procedure, we found and fixed a bug in the converter routines. However, since the quality of the output speech still seems inferior to that obtained by ISI, we believe that there are still compatibility problems in the code. A useful tool for finding such problems would be a file-to-file facility in FUD, such that the vocoder can be made to operate with speech input from a file of digitized speech and create a similar output file. With such a facility, identical input files could be used by both the ISI and BBN systems, and the resultant parameters could be compared. Furthermore, only in this way can standardized input files be used to test modifications to the vocoder program, in order that the effects on output speech quality can be systematically evaluated. We plan to implement such a facility, operating under

APEX (a FORTRAN-compatible, AP-120B executive, which runs on the PDP-11), in order to test the modifications we will be making to the vocoder algorithms.

3.1.4 Initial Network Experiment

We have tested the vocoder system by carrying on a conversation with ISI over the ARPA Network, and we have also used the "echo" facility of the vocoder program to listen to ourselves over the network. This testing procedure has pointed out a potentially serious problem. The PDP-11/40 seems unable to keep up with the real-time speech requirements. Because ISI uses a PDP-11/45, a significantly faster machine, they have not noticed this problem. We shall continue to investigate this problem during the next quarter. It is possible that the problem results simply from another configuration bug, but it is also possible that the solution will require substantial streamlining of the PDP-11 code.

3.2 Algorithm Modification Development Plan

In this section, we describe briefly our plan for modifying the AP-120B implementation of the LPC-II vocoder done at ISI and now operating at BBN. The goal of these modifications is to make the AP-120B program much more modular and thence amenable to substantial modification to improve the quality of the transmitted speech. We believe that this increased modularity and flexibility can be achieved with a minimal increase of computation time and memory requirement, at no jeopardy to the goal of real-time operation.

The ISI program structure is, as they have described it, "simple and straightforward, like that of a typical assembly language program for a minicomputer." Like most well-designed programs, its functional aspects are broken out into subroutines, such as windowing, autocorrelation, filtering, etc. However, like most assembly language programs, it is full of compiled-in address references and _ad hoc_ conventions about the form and locations of subroutine parameters and results. It is not particularly amenable to rearrangement and modification without careful attention to seeing that all relevant references in the source code are changed correctly.

A more flexible approach would be to clean up the loose ends by standardizing the passing of arguments to subroutines, the placement of results, and the use of data pad and scratch pad registers between subroutine calls. Such a standardization is already provided by the FPS AP Executive (APEX). We plan to adopt the APEX standard and to transform the ISI program into a set of APEX-callable program modules, callable from a host PDP-11 FORTRAN program.

The first step will be to break the vocoder program into only two APEX-callable modules, XANAL and XSYNTH, and to develop a FORTRAN host program to carry out proper LPC-II vocoding identical to the present ISI back-to-back implementation. The next step will be to subdivide the XANAL and XSYNTH routines further into routines that are comparable to the present ISI subroutines. The host program will then have to be transformed into a much more extensive set of APEX calls. At this point, most of the AP-120B vocoder program will have been broken into elementary functions.

We expect that this APEX transliteration of the ISI vocoder program will not run in real time, even with the speed enhancements made to APEX during this quarter (see Sec. 3.3.1 below), because of the overhead associated with starting each APEX routine. Therefore, it will only work in a non-real-time context, from and to files of digitized speech. Nor will the APEX vocoder transmit speech directly over the ARPANET, since it will be running under the control of FORTRAN under the RT-11 operating system, not the ISI-developed EPOS system. EPOS does not support FORTRAN.

However, the goal of real-time operation will be achievable by means of taking the set of APEX calls from the FORTRAN host program and stringing them together into a single AP-120B routine, using the Vector Function Chainer, a program development program available from FPS, which we intend to acquire. By moving the routine calling and argument passing from the PDP-11 to the much more rapid AP-120B, the overhead will be decreased to the point where real-time operation will again be feasible.

The goal of real-time network use will also be achievable, since the new single-module AP-120B vocoder routine will function quite similarly to the present AP-120B implementation, and it will be a trivial matter to interface it to the FUD and SPEECH programs presently operating under EPOS at ISI and BBN.

At this point, we will have changed the present ISI AP-120B vocoder implementation into a dual-purpose implementation, one purpose being flexible (but non-real-time) experimentation, the other purpose being real-time operation with the ARPANET. From this basis, we will be in a sound position to proceed with implementation of our previously developed advanced speech compression algorithms in the context of a real-time system. We have discussed these plans with the group at ISI, and they concur in their soundness and advisability. Our plans for the next quarter involve doing the modifications and proceeding with the implementation of the new algorithms.

3.3 Support Software

As part of the real-time vocoder development effort, we have built several pieces of support software for using the FPS AP-120B and the PDP-11/40 systems.

3.3.1 APEX and APLINK

The most general system for using the AP-120B for array processing is the AP Executive (APEX) supplied by Floating Point Systems Inc. APEX allows a host (PDP-11) FORTRAN program to transfer data to and from the AP-120B and to specify computational operations to be performed on the data by the AP-120B. The APEX library (APLIB) includes an extensive repertoire of elementary computational operations, and the user may program his own functions, to be used in the same way.

The APEX system, as delivered by FPS, is written in FORTRAN, which on our PDP-11 operating system (RT-11) produces inefficient code. As a result, each APEX routine called from the Host FORTRAN program consumes 2 to 4 ms of PDP-11 time just to start the computation process in the AP-120B. Many of these elemental operations require on the order of 1 ms to run in the AP-120B, so most of the time for running a multicall APEX program is spent in PDP-11 APEX overhead.

Analysis of the operations of APEX in starting an AP-120B computation showed that efficiency could be gained from (1) recoding the time-critical sections of APEX in machine language to avoid inefficient FORTRAN code and to eliminate the time-consuming nesting of FORTRAN subroutine calls and (2) streamlining the manner in which APEX is called by each individual user-called routine, once again eliminating needless nesting of FORTRAN subroutine calls. The latter was accomplished by modifying APLINK, one of the AP-120B program development programs. The resulting modified versions of these programs, APEX2 and APLNK2, were then tested, and the overhead associated with an APEX call was found to have been reduced to 0.5 to 1.0 ms.

3.3.2 SPS-FPS Support

In the past quarter we have developed and tested a package of PDP-11 subroutines that control transfers between the SPS-41 converters and the AP-120B Main Data memory. These subroutines automatically load SPS-41 program memory, initialize the sampling rate counters, and manage the double buffering required for real-time operation. They also return input and output data magnitude information to the calling program and indicate whether the real-time constraints have been met, and, if not, how much additional time was required. These subroutines have been designed to operate under APEX, the FPS-supplied executive.

3.3.3 IMSYS

Graphic displays are an invaluable aid to signal processing research. Since 1971, we have been using an IMLAC PDS-1 display computer in conjunction with our PDP-10 systems for interactive graphics support of our speech and signal processing research. A major part of the support software for the use of this graphics system has been IMSYS, a general-purpose graphics control program for the IMLAC, which creates and maintains displays specified by user programs operating in the host PDP-10. IMSYS graphics packages that are usable by user programs written in INTERLISP, FORTRAN, and BCPL have put interactive graphics at the disposal of a wide variety of user programs.

During the past quarter, we have extended the usability of IMSYS graphics to FORTRAN programs running on our Speech Processing PDP-11/40. The IMLAC part of IMSYS is used without modification. The host IMSYS graphics package was taken largely intact from another BBN project that had implemented IMSYS in RSX-11D FORTRAN; only new host-to-IMLAC data exchange routines and minor housekeeping changes were necessary. We will now be able to use IMSYS in support of the vocoder implementation effort on the PDP-11/40.

4. REFERENCES

1. J. Makhoul, C. Cook, R. Schwartz, and D. Klatt, "A Feasibility Study of Very Low Rate Speech Compression Systems," Report No. 3508, Bolt Beranek and Newman Inc., Cambridge, Mass., Feb. 1977.

2. J. Makhoul and R. Viswanathan, "Adaptive Lattice Methods for Linear Prediction," 1978 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Tulsa, Okla., April 10-12, 1978, pp. 83-86.

3. J. Makhoul, R. Viswanathan, R. Schwartz, and A.W.F. Huggins, "A Mixed-Source Model for Speech Compression and Synthesis," 1978 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Tulsa, Okla., April 10-12, 1978, pp. 163-166.

4. R. Schwartz, "Acoustic Phonetic Experiment Facility for the Study of Continuous Speech," 1976 Int. Conf. on Acoustics, Speech, and Signal Processing, Philadelphia, Pa., April 12-14, 1976, pp. 1-4.

5. R. Viswanathan, J. Makhoul, and A.W.F. Huggins, "Speech Compression and Evaluation," Report No. 3794, Bolt Beranek and Newman Inc., Cambridge, Mass., April 1978a.

6. R. Viswanathan, W. Russell, and J. Makhoul, "Objective Speech Quality Evaluation of Narrowband LPC Vocoders," 1978 IEEE Conf. on Acoustics, Speech, and Signal Processing, Tulsa, Okla., April 10-12, 1978b, pp. 591-594.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER BBN Report No. 3896	2. GOVT ACCESSION NO. AD-A155468	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SPEECH COMPRESSION AND SYNTHESIS		5. TYPE OF REPORT & PERIOD COVERED Quarterly Technical Report 6 April to 5 July 1978
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Jared Wolf, John Makhoul, Lynn Cosell, Richard Schwartz, John Klovstad		8. CONTRACT OR GRANT NUMBER(s) F19628-78-C-0136
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 50 Moulton St. Cambridge, Mass. 02138		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Deputy for Electronic Technology (RADC/ETC) Hanscom Air Force Base, Mass. 01731 Contract Monitor: Mr. Caldwell P. Smith		12. REPORT DATE July 1978
		13. NUMBER OF PAGES 36
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse; Department of Commerce for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 3515.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Speech synthesis, phonetic synthesis, diphone, LPC synthesis, vocoder, speech compression, linear prediction

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This document reports progress in the development of a phonetic speech synthesis algorithm and in the implementation of a real-time vocoder. The synthesis algorithm uses LPC synthesis from the concatenation of diphone templates. The methodology and basic software of the synthesis work are described. In vocoder development, the installation of the complete ISI LPC-II vocoder system on our PDP-11 and AP-120B computers is described, as is our plan for the modification of this system for improved speech quality.